Bridging the gap between Ox and Gauss using OxGauss

Bridging the Gap Between Ox and Gauss using OxGauss

Sébastien Laurent

CeReFim (Université de Namur)

and CORE (Université catholique de Louvain).

E-mail: Sebastien.Laurent@fundp.ac.be.

Correspondence to CeReFim, 8 rempart de la vierge, B5000 Namur, Belgium.

Phone: +32 (0) 81 724869.

Fax: +32 (0) 81 724840.

and

Jean-Pierre Urbain

Department of Quantitative Economics, University Maastricht,

The Netherlands. E-mail: j.urbain@ke.unimaas.nl.

First Draft: October 2003

This version: January 2004

1 Introduction

The purpose of this paper is to review and discuss the key improvements brought to OxGauss, a program that comes with modern versions of Ox.[1] OxGauss provides a way to run Gauss[2] code under Ox or to call an existing Gauss procedure under Ox in a similar way than one can call C (dll) or fortran codes from Gauss and Ox. Unlike the old g2ox program that came with earlier versions of Ox, OxGauss is *not* a translator from Gauss to Ox.

Depending on the goal of the analysis and on the experience of the user, both use can prove to be particularly useful. From an Ox user point of view, the main objective of OxGauss is probably to allow the many existing Gauss codes to be called from Ox with only a minimum number of changes

[1]There are two versions of Ox. Oxconsole can be downloaded from *http://www.nuff.ox.ac.uk/Users/Doornik/*, which is the main Ox web page. The console version is free for educational purposes and academic research. The Professional Windows version, or commercial version comes with a nice interface for graphics known as GiveWin (available for purchase from Timberlake Consultants, *http://www.timberlake.co.uk*).

[2]Gauss is sold by Aptech Systems, 23804 S.E. Kent-Kangley Rd., Maple Valley, WA, 98038, USA; see *http://www.aptech.com/*.

to these codes. This is beneficial to both the Ox user and the writer of the Gauss codes as it increases the visibility and hence the potential use of the underlying statistical technique. This may also help in the transition from Gauss to Ox if this is the purpose of the exercise.

On the other hand, running a pure Gauss code with OxGauss is attractive for the non-Gauss and potentially even non-Ox users in that it makes the replication of published work possible using the console version of Ox. It has considerable merit since the replicability of simulation and empirical results in econometrics is recognized as being a fairly important aspect of research as exemplified by the practice of the *Journal of Applied Econometrics* that asks authors to make their data and possibly specialized codes available to the potentially interested reader. Not surprisingly, an increasing number of researchers in econometrics are making their codes and routines freely available to the econometrics community. For this reason, OxGauss also fulfils a genuine need in that it provides the researcher with a free and rather simple solution to run Gauss codes.

The paper is structured as follows: in Section 2, we propose an overview of OxGauss and give some simple examples as well as a speed comparison between Ox, OxGauss and Gauss 3.5. Section 3 discusses the graphical issue. Section 4 proposes to test the usefulness of OxGauss in replicating the results of a broad number of research papers. Finally, Section 5 concludes.

2 OxGauss

As explained above, the purpose of OxGauss is twofold: calling Gauss codes from Ox and running Gauss codes without having to install Gauss. The next two subsections illustrate these two features of OxGauss.

2.1 Calling Gauss codes from Ox

The main objective of OxGauss is probably to allow Gauss code to be called from Ox. This helps in the transition to Ox, and increases the amount of code that is available to users of Ox.

In order to illustrate how Gauss codes can be called from OX, we consider a small project that mixes both Gauss and Ox codes. The first file, *Gaussprocs.src*, consists of a code file containing the procedure *gengarch(omega,alpha,beta,nu,T_0,T,n)* that simulates a GARCH model. This procedure has been written by Dick van Dijk (see Franses and van Dijk, 2000) and is downloadable from his web site `http://www.few.eur.nl/few/people/djvandijk/nltsmef/nltsmef.htm`.

To call this procedure from an Ox code, one first has to create a header file. The purpose of the header file is simply to allow the declaration of the functions, constants and external variables to be known wherever it is required. The reason is that, to avoid a compilation error in Ox, any function or global variable has to be explicitly declared before its use. In our example, the header file (*Gaussprocs.h*) consists of the following instructions:

Gaussprocs.h

```
#include <oxstd.h>
namespace gauss
{
    gengarch(const omega,const alpha,const beta,const nu,const T_0,
             const T, const n);
    // Add new procedures here
}
```

Additional procedures can be added in *Gaussprocs.src* but the header file has to be modified accordingly.[3] It is recommended to use the *.src* extension for the Gauss programs and *.h* for the header files.

In the example *GarchEstim.ox*, we use the Gauss procedure to generate 20.000 observations following a GARCH(1,1) process with Student-t errors and then, rely on the Ox package G@RCH 3.0 (see Laurent and Peters, 2002) to estimate a GARCH(1,1) model by gaussian Quasi-Maximum likelihood. To do so, the OxGauss code must be imported into the Ox program, together with the G@RCH package. The **#import** command has been extended to recognize OxGauss imports by prefixing the file name with `gauss::`.

GarchEstim.ox

```
#include <oxstd.h>
#import <packages/Garch30/garch>
#import "gauss::Gaussprocs"                                        <---
main()
{
    decl omega=0.2; decl alpha=0.1; decl beta=0.8; decl nu=10;
    decl T_0=1000; decl T=20000; decl n=1;
    decl y=gauss::gengarch(omega,alpha,beta,nu,T_0,T,n);           <---
    decl garchobj;
    garchobj = new Garch();
    garchobj.Create(1, 1, 1, T, 1);
    garchobj.Append(y, "Y");
    garchobj.Select(Y_VAR, "Y",0,0 );
    garchobj.SetSelSample(-1, 1, -1, 1);
    garchobj.DISTRI(0);             //0 for Normal
    garchobj.GARCH_ORDERS(1,1);     //p order, q order
    garchobj.MODEL(1);              //1: GARCH
    garchobj.Initialization(<>);
    garchobj.DoEstimation();
    garchobj.Output();
    delete garchobj;
}
```

Note that when the OxGauss functions or variables are accessed, they must also be prefixed with the identifier `gauss::`.

To run this program on the command line, the user simply has to enter `oxl GarchEstim.ox`. Alternatively, it can be launched from OxEdit. OxEdit 1.62 (or later) is a free but powerful text editor for Windows provided with both versions of Ox 3.3. Like GiveWin, OxEdit supports syntax colouring

[3] Arguments declared **const** can be referenced, but cannot be changed inside the function.

of Ox programs, and context-sensitive help. The first time OxEdit is used, the user should execute the `Preferences/Add Predefined Modules` menu and select Ox. Ox and Gauss programs can then be run from the Modules menu without leaving OxEdit. See also the OxEdit web page `http://www.oxedit.com` for more details. Finally, users of the Ox Professional can run Ox codes within GiveWin by using the menu `Modules/Start OxRun`.

2.2 Running Gauss codes

The second potentially attractive feature of OxGauss is to enable the user to directly run a wide range of Gauss programs under Ox.

As an example, we consider the Gauss package *Mixed Logit Estimation Routine for Panel Data* of Kenneth Train, David Revelt and Paul Ruud. The archive file *train0299.zip* (available at `http://elsa.berkeley.edu/Software/abstracts/train0296.html`) contains seven files including the code file `mxlp.g` and the data. This program has been written by Kenneth Train and used by this author in a collection of papers (see the web site above for more details) dealing with mixed logit models.[4]

In order to save space, we do not report the 1396 lines of code of the main file `mxlp.g`. This program can be run on the command line by entering `oxl -g mxlp.g`. Alternatively, it can be launched from OxEdit (`Modules/OxGauss` menu) or within GiveWin by using the menu `Modules/Start OxGauss`. Note that while the previous versions of OxGauss required a few modifications on the codes,[5] the program is now almost fully compatible with the new version. The only problem is that the program estimates the model by maximum likelihood, giving the opportunity to the user to chose the maximization routine domax of Paul Ruud or the commercial package maxlik. Launching the program could lead to the following error message:

```
path...\mxlp.g (1372): maxlik file not found
path...\mxlp.g (1373): maxlik.ext include file not found
```

To solve this problem one can either comment out lines 421 to 451 concerning the add-on maxlik (and use the domax procedure, i.e. the default option $OPTIM = 1$ in the program) or install the M@ximize package used in Section 4. As expected, the results are very similar, if not identical (the only difference is detected after the sixth decimal of the standard errors).

[4] Mixed logit (also called random-parameters logit) generalizes standard logit by allowing the parameter associated with each observed variable (e.g., its coefficient) to vary randomly across units (e.g. individuals or customers).

[5] On his web site (*http://facweb.arch.ohio-state.edu/pviton/support/oxgauss*), Philip Viton mentioned about 6 operations to make under Ox 3.2 before succeeding to run the code without compilation error.

2.3 Understanding OxGauss

When an OxGauss program is run, it automatically includes the `/include/oxgauss.ox` file. This itself imports the required files:[6]

/include/oxgauss.ox

```
#define OX_GAUSS
#import <g2ox>
#import <gauss::oxgauss>
```

These import statements ensure that `g2ox.h` and `oxgauss.h` are being included. The majority of the OxGauss run-time system is in `\include\g2ox.ox` while the keywords are largely in `oxgauss.src`.

A nice feature of OxGauss is its transparency since most of the codes that link Gauss functions to Ox are gathered in the file `\include\g2ox.ox`. For instance, the output of the Gauss function `cumprodc(x)` is a $N \times K$ matrix with the cumulative products of the columns of the $N \times K$ matrix `x`. The Ox code here below (copy from the file `g2ox.ox`) shows how OxGauss interprets this function.

part of g2ox.ox

```
cumprodc(const mx)
{
    return ::cumprod(mx);
}
```

From this example, it is clear from that OxGauss does not translate the Gauss code into Ox but makes the link between the Gauss function (here `cumprodc`) and its Ox counterpart (`cumprod`). When the corresponding Ox function does not exist, an ox code is written between the brackets, aiming at computing the same thing as the original Gauss function. It is important to note all the Gauss functions are not supported by OxGauss. For instance, there is no equivalent of the Gauss function `intgrat2` (for the computation of double integrals) in Ox 3.3. For this reason, the corresponding procedure in `g2ox.ox` just reports the error message `intgrat2() unsupported` (see below). However, if such a function becomes available in a next version of Ox, mapping `ingrat2` to the corresponding function in Ox will be a child's play!

Tables A1 and A2 of the appendix (available on the web site of the JAE Data Archive) give a list of all the Gauss functions supported by OxGauss. To simplify the reading of the list, we report pre-compiled functions (or directly mapped functions) like `sin` in Table A1 and open source functions (like `cumprodc`, see above) in Table A2. Adding adding all the functions leads to a total of 420 functions recognized by OxGauss. Table A3 in the appendix gives a list of 64 Gauss functions not supported by the current version of Ox (or about 15%).

[6]For ease of presentation, the filename is printed in the upper right corner of the window.

2.4 Speed Comparison

As pointed out by Cribari-Neto (1997), *the main strength of Ox is its speed*, although Gauss performs quite well too and its speed performance are not far behind those of Ox. A recent and detailed comparison between several mathematical programs performed by Stefan Steinhaus (see `http://www.scientificweb.de/ncrunch/`) shows that Ox is the winner in terms of speed. Since OxGauss just implements a layer on Ox, OxGauss is expected to be slower than Ox. But one may wonder how slower is it and how it really compares to Gauss in terms of speed? To answer these two questions we consider the Benchmark tests proposed by Stefan Steinhaus (edition 3). Note that since the functions *intquad2* and *intquad3* (double and triple integration of functions) are not available in Ox 3.3, the corresponding tests have been discarded which leads a total of 14 points of comparison. To perform the speed comparison, we did first execute the Ox benchmark program `Benchox2.ox` with 5 replications of each test on a Pentium III 450 Mhz and 526 MB RAM running under Windows 98 with Windows versions of the programs. We also conducted the same experiment with the Gauss benchmark program `Benchga2.prg` using OxGauss, Gauss 3.5. The results are reported in Table 1.

Table 1 Speed Comparison (timing in seconds) between Ox 3.3, OxGauss and Gauss 3.5.

Operation	Ox 3.3	OxGauss	Gauss 3.5
Creation, trans. & reshaping of a 1000x1000 matrix:	1.043	1.153	1.043
1000x1000 random matrix to the power 1000:	1.023	1.003	1.083
Sorting of 2,000,000 random values:	9.190	9.577	9.643
FFT over 1,048,576 random values:	4.777	5.423	5.417
Determinant of a 1000x1000 random matrix:	14.590	14.593	14.593
Creation of an 1400x1400 Toeplitz matrix:	0.167	0.167	0.200
Inverse of a 1000x1000 random matrix:	36.433	36.600	36.930
Eigenvalues of a 600x600 random matrix:	35.573	35.867	36.563
Choleski decomposition of a 1000x1000 random matrix:	4.360	4.380	4.437
Creation of 1000x1000 cross-product matrix:	8.953	12.817	12.777
Calculation of 500000 fibonacci numbers:	1.377	1.380	1.423
Gamma function on a 1000x1000 random matrix:	0.737	0.763	0.730
Gaussian error function over a 1000x1000 random matrix:	0.930	0.950	0.790
Linear regression over a 1000x1000 random matrix:	28.563	28.597	28.713

Benchmark programs run (5 replications of each test) on a Pentium III 450 Mhz.

As a whole, we see from this table that OxGauss compares very well to Gauss 3.5 in terms of speed. Indeed, while Ox is in general faster than OxGauss and Gauss 3.5 (when taking the 14 experiments we see that Ox is about 4% faster than OxGauss and Gauss 3.5), the difference between OxGauss and Gauss 3.5 is very small (OxGauss is found to be on average about 0.5% faster than

Gauss 3.5). [7]

3 Graphs using GnuDraw

While most graphical features of Gauss are recognized by OxGauss via Ox Professional for Windows (through GiveWin), Oxconsole has no graphs support. Nevertheless, the user of the console version can rely on the Ox package GnuDraw developed by Charles Bos that allows the creation of GnuPlot (see `http://www.gnuplot.info`) graphics from Ox. The package is free of charge and is downloadable from his homepage `http://www.tinbergen.nl/~cbos/`, together with the GnuPlot software. Note that a detailed help file *gnudraw.html* and a few example codes are provided with the package, which makes its use very friendly.

A nice feature of this package is its platform independence (unlike the current version of GiveWin which is only available for Windows) which provide non-Windows users (e.g. Unix, Linux, Solaris, Sun, etc.) an efficient graph support. When using Ox 3.30, GnuPlot can be called automatically from within Ox. Usage of GnuDraw is intended to be simple - see Cribari-Neto and Zarkos (2003) for a comprehensive overview of the GnuDraw package. Interestingly, as OxGauss implements just a layer over Ox, it is possible to instruct the underlying Gauss to call GnuDraw routines instead of the OxDraw routines. The program `gnuGauss.prg` implements this.

gnuGauss.prg

```
library pgraph;

x=seqa(1,1,1000);
y=rndn(1000,1);
xlabel("X-axis");
ylabel("Y-axis: Normal(0,1) draws");
call xy(x, y);

end;
```

4 Replicating empirical results using OxGauss

The main drawback of OxGauss until now is that it was not suited to run Gauss codes that make use of commercial applications. However, an interesting characteristic of OxGauss is that it is extensible. The GnuDraw package presented here above is a first example of additional package extending OxGauss.

For instance OxGauss reports an error message if the Gauss code requires one of the optimizators from the modules Cml, Maxlik, or Optmum (see Section 2.2). This makes OxGauss useless in many

[7] While a comparison with a more recent version of Gauss is of potential interest, we do not investigate this issue since the speed comparison is not the aim of the paper. Note that we have done the same experiment with Gauss 3.2.29 and the result is that Gauss 3.5 is a much improved release in terms of speed.

situations of practical interest, ...except if we provide to Oxgauss an additional package that bridges the gap between these three libraries and the optimizators of Ox.

To test the feasibility of this task, we have written a set of procedure gathered in a package called M@ximize 1.0. By means of transparency, the package is open source and freely available on the web at the following address: `http://www.core.ucl.ac.be/˜laurent`.

It is important to note first that it is not our intention to review our package here. Indeed, the main goal of this section is to test the reliability OxGauss in a non-trivial situation. Second, the M@ximize package *does not translate* the Gauss optimizators into Ox as we just link the options; nor to clone the optimizators. Indeed, in the first release of the package (M@ximize 1.0) not all the functions of these optimizators are available, although the most important options of Cml, Maxlik, and Optmum are taken into account.

As explained above, the main reason for using OxGauss is probably to replicate the results obtained in a research paper. To test OxGauss in a real-life situation, we have downloaded from the internet a huge number of Gauss codes. Here is a list of five web sites that we have visited from which both the data and the Gauss codes can be retrieved:
James Hamilton: `http://weber.ucsd.edu/˜jhamilto/`
Bruce Hansen: `http: //www.ssc.wisc.edu/˜bhansen/`
Chang-Jin Kim: `http://www.econ.washington.edu/user/cnelson/SSMARKOV.htm`
Luc Bauwens: `http://www.core.ucl.ac.be/econometrics/bauwens.htm`
Rolf Tschering: `http://www.personeel.unimaas.nl/r.tschernig/`

We have also used the codes provided by Kim and Nelson (1999) in their book on markov-switching models (Chapters 3 to 11). Table A4 in the appendix gives the list of papers that we have replicated (i.e. 27 references). Most of these papers rely quite heavily on non-linear optimization techniques and thus require one the three above mentioned optimizators of Gauss. Consequently, to run these codes under OxGauss one need first to install the M@ximize package (i.e. by unzipping the file *m@ximize.zip* in the main directory of Ox).

While most of the codes can be run in their present form, some marginal changes in the GAUSS codes are sometimes needed. The most frequently encountered problems are:

- *Converting data files*. For instance, running the Gauss code related to the reference 22 in Table A4 gives the Invalid .FMT or .DAT file error message. The reason is that old style Gauss data sets (v89 *.dht/.dat*) must be converted to the new Gauss format (v96 *.dat*). The program to do this conversion is ox/lib/dht2dat. The conversion can be run from the command line as:

  ```
  oxl lib/dht2dat old_datafile.dht new_datafile.dat
  ```

  Alternatively, the data files can be converted to the new format through GiveWin by loading first the *.dht* file and second saving the file into the new format.

- *Absence of extension.* To launch a Gauss code using OxEdit, the file needs an extension. It is common to use the extension *.src*.
- *Interactive mode.* Examples 1, 6, 9 and 13 use the Gauss function `cons` that requests an input from the keyboard (console) and puts it into a string. The typical use of this function is to generate a message like "Do you wish to continue (y or n)?". Depending of the result the program takes one direction or the another. In other words, the program enters in an interactive mode. In such a case the program has to be launched using "Ox interactive", i.e. Oxli.exe under Windows instead of Oxl.exe.[8]

As illustration, we consider the Gauss package written by Rolf Tschernig accompanying the paper of Yang and Tschernig (1999) published in the *Journal of the Royal Statistical Society, Series B* (reference 27 in Table A4). We focus on the example file provided by the author, i.e. *multband.tes* that estimates the asymptotic optimal vector bandwidth for simulated bivariate non-linear regression models. This file is made up of about 190 lines of Gauss code and includes three libraries, i.e. Optmum, pgraph and multband (a library provided by the author) as well as a set of three dll files. To use the package under Gauss, one has first to install the library multband by first copying the file *multband.lcg* into the subdirectory `./lib` of Gauss and second the files *multband.src* (about 2900 lines of code) and *multband.dec* (declarations of global variables) into the subdirectory `./src`. Finally, to complete the installation, one has to copy the three dll files *locling.dll, density.dll* and *loccubg.dll* into the subdirectory `./dlib`. Importantly, to use the package under OxGauss, one has to follow the same instructions and copy the files into the existing subdirectories `./OxGauss/lib`, `./OxGauss/src` and `./OxGauss/dlib`.

Now the example file *multband.tes* can be executed.[9] Once again, the results are identical (up to the sixth decimal). The outputs obtained with OxGauss and Gauss 3.2 are reported in Table A5 (in the appendix).

5 Conclusion

This paper presents a review and a discussion of OxGauss, an application that enables the user to run a wide range of Gauss programs/codes under Ox without needing to have Gauss installed on his/her machine. One main drawback of OxGauss is that it is of little use once the purpose is to execute a program that requires one of the three well know Gauss application modules Cml, Maxlik or Optmum.

[8] When using OxEdit to run the Gauss code, an additional shortcut has to be created. The simple solution is to click on the menu VIEW/PREFERENCES/ADD/REMOVE MODULES. Then clone the OxGauss shortcut and in the Command line change Oxl.exe by Oxli.exe.

[9] Notice that this example file simulates a sequence of 250 observations. To make the comparison between Gauss and OxGauss possible we have changed the original code that now always uses (load) the same random numbers.

In this paper we show that since OxGauss is extensible, it is possible to fix this problem by writing a set of additional procedures that bridge the gap between Ox and the above mentioned optimizers.

It is important to emphasize that OxGauss is potentially useful both for Gauss and Ox users. On the one hand Gauss codes can efficiently be called under Ox. This means that Ox programmers willing to use existing Gauss procedures do not have to translate the procedures into Ox but can call the Gauss code directly under Ox. On the other hand, OxGauss can be used to run the Gauss code(s) under Ox and hence to replicate the results of a paper for which Gauss code is made available by the author(s). The effectiveness of OxGauss is illustrated by revisiting a large number of Gauss codes that are freely available on the internet and that use Gauss application modules that require numerical optimization (26 papers published in international reviews and the procedures related to a book, see Table A4). Importantly in all cases the programs were found to be fully compatible with OxGauss in the sense that no change was required (or very minor changes) on the original code and that the results were almost identical.

To conclude, we believe that OxGauss could bring the Gauss and Ox user communities closer. We could even hope that Gauss users who already share their Gauss codes would start testing the compatibility with OxGauss, make the changes to ensure compatibility if necessary and indicate that their code is "OxGauss compliant".

6 Acknowledgment

While remaining responsible for any errors in this paper, the authors would like to thank the editor James G. MacKinnon and Jean-Philippe Peters for useful comments and Charles Bos and Jurgen Doornik for their accessibility.

References

CRIBARI-NETO, F. (1997): "Econometric Programming Environments: Gauss, Ox and S-Plus," *Journal of Applied Econometrics*, 12, 77–89.

CRIBARI-NETO, F., and S. ZARKOS (2003): "Econometric and Statistical Computing Using Ox," *Computational Economics*, 21, 277–295.

FRANSES, P., and D. VAN DIJK (2000): *Non-Linear Series Models in Empirical Finance*. Cambridge University Press.

KIM, C.-J., and C. NELSON (1999): *State-Space Models with Regime Switching: Classical and Gibbs-Sampling Approaches with Applications*. The MIT Press.

LAURENT, S., and J.-P. PETERS (2002): "G@RCH 2.2 : An Ox Package for Estimating and Forecasting Various ARCH Models," *Journal of Economic Surveys*, 16, 447–485.

Appendix of

Bridging the Gap Between Ox and Gauss using OxGauss

Sébastien Laurent

CeReFim (Université de Namur)

and CORE (Université catholique de Louvain).

E-mail: Sebastien.Laurent@fundp.ac.be.

Correspondence to CeReFim, 8 rempart de la vierge, B5000 Namur, Belgium.

Phone: +32 (0) 81 724869.

Fax: +32 (0) 81 724840.

and

Jean-Pierre Urbain

Department of Quantitative Economics, University Maastricht,

The Netherlands. E-mail: j.urbain@ke.unimaas.nl.

First Draft: October 2003

This version: January 2004

Table A1: Precompiled functions supported by OxGauss

_fcmptol	delete	lib library	output	save
abs	det	library	outwidth n	saveall
arccos	diagrv	line	pdfn	screen
arcsin	disable	ln	plot x,y	scroll
arctan	dlibrary	load x	plotsym n	shell
arctan2	dllcall	loadf f	pqgwin	show
atan	enable	loadk k	prcsn n	sin
atan2	end/stop	loadm x	print	sinh
cdfchic	erf	loadp p	printdos str	sqrt
cdfchii	errorlog str	loads s	rank	system
cdffc	exp	locate m,n	replay	tan
cdfn	eye	log	rerun	tanh
cdfnc	feq	lowmat	rev	toeplitz
cdfni	floor	lprint	rndcon c	trace new
cdftc	fmod	lpwidth n	rndmod m	trap new
ceil	format	lshow	rndmult a	trunc
cols	gamma	meanc	rndn	use gcgfile
cos	graph	median	rndseed seed	vcx
cosh	hsec	msym str	rndu	vech
create	inv	new	round	xpnd
datalist	invpd	ones	rows	zeros
debug	ismiss	open	run filename	

Table A2: Open source functions supported by OxGauss

balance	corrvc	etstr	keyw	polymult	seekr	upper
band	corrx	exctsmpl	lag1	polyroot	selif	utrisol
bandchol	counts	exec	lagn	printfm	seqa	vals
bandcholsol	countwts	export	lncdfbvn	printfmt	seqm	vcm
bandltsol	crossprd	exportf	lncdfn	prodc	setcnvrt	vec
bandrv	crout	fcheckerr	lncdfn2	putf	setdif	vecr
bandsolpd	croutp	fclearerr	lncdfnc	qnewton	setvmode	vget
base10	csrtype	fflush	lnfact	qprog	shiftr	wait
besselj	cumprodc	fft	lnpdfmvn	qqr	sleep	waitc
bessely	cumsumc	ffti	lnpdfn	qqre	solpd	writer
cdfbeta	cvtos	fftn	loadd	qqrep	sortc	xpnd
cdfbvn	date	fge	lower	qr	sortcc	
cdfchinc	datestr	fgets	lowmat1	qre	sorthc	
cdffnc	datestring	fgetsa	ltrisol	qrep	sorthcc	
cdfgam	datestrymd	fgetsat	lu	qrsol	sortind	
cdfn2	dayinyr	fgetst	lusol	qrtsol	sortindc	
cdftci	delif	fgt	maxc	qtyr	sortmc	
cdftnc	design	files	maxindc	qtyre	sqpsolve	
cdftvn	detl	fle	maxvec	qtyrep	stof	
cdir	dfft	flt	mbesselei	quantile	stop	
changedir	dffti	fne	mbesselei0	qyr	strindx	
chol	dfree	fopen	mbesselei1	qyre	strlen	
choldn	diag	formatcv	mbesseli	qyrep	strput	
cholsol	dos	formatnv	mbesseli0	rankindx	strrindx	
cholup	dotfeq	fputs	mbesseli1	readr	strsect	
chrs	dotfge	fputst	meanc	real	submat	
close	dotfgt	fseek	minc	recode	subscat	
closeall	dotfle	fstrerror	minindc	recserar	substute	
cls	dotflt	ftell	miss	recsercp	sumc	
cmadd	dotfne	ftocv	missex	recserrc	svd	
cmcplx	dstat	ftos	missrv	rfft	svd1	
cmcplx2	dummy	gammaii	moment	rffti	svd2	
cmdiv	dummybr	gausset	ndpchk	rfftip	svdcusv	
cmemult	dummydn	getf	ndpclex	rfftn	svds	
cmimag	eig	getname	ndpcntrl	rfftnp	svdusv	
cminv	eigh	gradp	null1	rfftp	sysstate	
cmmult	eighv	hasimag	ols	rndbeta	system	
cmreal	eigrg	hessp	olsqr	rndgam	tab	
cmsoln	eigrg2	imag	olsqr2	rndnb	tempname	
cmsub	eigrs	import	orth	rndns	time	
cmtrans	eigrs2	indcv	packr	rndp	timestr	
code	eigv	indexcat	parse	rndus	token	
color	end	indices	pause	rndvm	trapchk	
colsf	envget	indices2	pi	rotater	trim	
con	eof	indnv	pinv	rowsf	trimr	
cond	eqsolve	intrsect	polychar	rref	type	
cons	erfc	invswp	polyeval	save	union	
conv	error	iscplx	polyint	saved	uniqindx	
coreleft	etdays	iscplxf	polymake	scalerr	unique	
corrm	ethsec	key	polymat	scalmiss	upmat	

Table A3: Functions not supported by OxGauss (under Ox 3.3)

cdfbvn2	filesa	makevars	sortd	vnamecv
cdfbvn2e	getnr	medit	spline1d	vput
cdfmvn	getpath	mergeby	spline2d	vread
complex	header	mergevar	stdc	vtypecv
conj	hess	momentd	tocart	
csrcol	importf	nametype	topolar	
csrlin	intgrat2	nextn	typecv	
editm	intgrat3	nextnevn	typef	
eigcg	intquad1	null	varget	
eigcg2	intquad2	optn	vargetl	
eigch	intquad3	optnevn	varput	
eigch2	intrleav	quantiled	varputl	
fftm	intsimp	schtoc	vartype	
fftmi	lncdfbvn2	schur	vartypef	
fileinfo	lncdfmvn	setvars	vlist	

Table A4: List of codes associated to papers

1.	HAMILTON, J. (1994): *State-Space Models*, in Handbook of Econometrics, Volume 4, 3039–3080, edited by R.F. Engle and D., McFadden, Amsterdam: North Holland.
2.	HAMILTON, J. (1996): “The Daily Market for Federal Funds”, *Journal of Political Economy*, pp. 26–56.
3.	HAMILTON, J. (1996): “Specification Testing in Markov-Switching Time-Series Models”, *Journal of Econometrics*, 70, 127–157.
4.	HAMILTON, J., and C. ENGLE (1990): “Long Swings in the Exchange Rate: Are They in the Data and Do Markets Know It?”, *American Economic Review*, pp. 689–713.
5.	HAMILTON, J., and O. JORDA (2002): “A Model for the Federal Funds Rate Target”, *Journal of Political Economy*, 110, 1135–1167.
6.	HAMILTON, J., and G. LIN (1996): “Stock Market Volatility and the Business Cycle”, *Journal of Applied Econometrics*, 11, 573–593.
7.	HAMILTON, J., and G. PEREZ-QUIROS (1996): “What Do the Leading Indicators Lead?”, *Journal of Business*, 69, 27–49.
8.	HAMILTON, J., and R. SUSMEL (1994): “Autoregressive Conditional Heteroskedasticity and Changes in Regime”, *Journal of Econometrics*, 64, 307–333.
9.	Bauwens, L. M. Lubrano (1998): Bayesian Inference on GARCH models using the Gibbs Sampler, *The Econometrics Journal*, 1, C23-C46.
10.	Hansen, B. (1992): “Tests for Parameter Instability in Regressions with I(1) Processes”, *Journal of Business and Economic Statistics*, 10, 321-335.
11.	Hansen, B. (1992): “Testing for Parameter Instability in Linear Models”, *Journal of Policy Modeling*, 14, 517-533.
12.	Hansen, B. (1992): “The likelihood Ratio Test under Non-standard Conditions: Testing the Markov Switching Model of GNP”, *Journal of Applied Econometrics*, 7, S61-S82.
13.	Hansen, B. (1994): “Autoregressive Conditional Density Estimation”, *International Economic Review*, 35, 705-730.
14.	Hansen, B. (1996): “Inference when a Nuisance Parameter is not Identified under the Null Hypothesis”, *Econometrica*, 64, 413-430.
15.	Hansen, B. and A. Gregory (1996): “Residual-based Tests for Cointegration in Models with Regime Shifts”, *Journal of Econometrics*, 70, 99-126.
16.	Hansen, B. (1997): “Approximate Asymptotic p-values for Structural Change Tests”, *Journal of Business and Economic Statistics*, 15, 60-67.
17.	Hansen, B. (1997): “Inference in TAR Models”, *Studies in Nonlinear Dynamics and Econometrics*, 2, 1-14.
18.	Hansen, B. (1999): “Testing for Linearity”, *Journal of Economic Surveys*, 13, 551-576.
19.	Hansen, B. (2000): “Sample Splitting and Threshold Estimation”, *Econometrica*, 68, 575-603.
20.	Hansen, B. (2000): “Testing for Structural Change in Conditional Models”, *Journal of Econometrics*, 97, 93-115.
21.	Hansen, B. and M. Caner (2000): “Threshold Autoregression with a Unit Root”, *Econometrica*, 69, 1555-1596.
22.	Hansen, B., D. Cox and E. Jimenez: “How Responsive are Private Transfers to Income? Evidence from a Laissez-faire Economy”, forthcoming in *Journal of the Public Economics*.
23.	Hansen, B. and B. Seo (2002): “Testing for Threshold Cointegration”, *Journal of Econometrics*, 110, 293-318.
24.	Hansen, B. (2001): “The New Econometrics of Structural Change: Dating Changes in U.S. Labor Productivity”, *Journal of Economic Perspectives*, 15, 117-128.
25.	Hansen, B.: “Recounts from Undervotes: Evidence from the 2000 Presidential Election”, forthcoming in *Journal of the American Statistical Association*.
26.	Kim, C.-J. and C. Nelson (1999): *State-Space Models with Regime Switching: Classical and Gibbs-Sampling Approaches with Applications*, The MIT Press.
27.	Yang, L. and R. Tschernig (1999): “Multivariate Bandwidth Selection for Local Linear Regression”, *Journal of the Royal Statistical Society, Series B*, 61, 793-815.

Table A5: Outputs obtained by running *multband.tes* (see reference 27 in Table A4) under OxGauss (left) and Gauss 3.2 (right)

OxGauss (left)	Gauss 3.2 (right)
Ox version 3.30 (Windows) (C) J.A. Doornik, 1994-2003	
hdrot_ll: chosen block: 2.0000000 1.0000000	hdrot_ll: chosen block: 2.0000000 1.0000000
Results from bandrot.g	Results from bandrot.g
h_ROT	h_ROT
0.051626512	0.051626512
0.051626512	0.051626512
hd_ROT	hd_ROT
0.044616931	0.044616919
0.064029894	0.064029901
B_hat 0.23183482	B_hat 0.23183482
Cm_hat	Cm_hat
5535.4077 477.33950	5535.4077 477.33950
477.33950 1305.0202	477.33950 1305.0202
C_hat 7795.1069	C_hat 7795.1069
hdrot_ll: chosen block: 2.0000000 1.0000000	hdrot_ll: chosen block: 2.0000000 1.0000000
hcdrotlp: chosen block: 2.0000000 1.0000000	hcdrotlp: chosen block: 2.0000000 1.0000000
hcdrotlp: Blamu	hcdrotlp: Blamu
78617.709 -3421.7567	78617.709 -3421.7567
-4919.6541 80.592526	-4919.6541 80.592526
Results from bandpi.g	Results from bandpi.g
h_PI	h_PI
0.076176271	0.076176271
0.076176271	0.076176271
hd_PI	hd_PI
0.082851089	0.082851007
0.064612911	0.064612978
Bd_hat 0.37280023	Bd_hat 0.37280024
hd_ROT	hd_ROT
0.044616931	0.044616919
0.064029894	0.064029901
C_hat 1227.6312	C_hat 1227.6312
hC_ROT 0.18391889	hC_ROT 0.18391889
Cm_hat	Cm_hat
458.95790 18.900462	458.95789 18.900464
18.900462 1240.7565	18.900464 1240.7565
hCd_ROT	hCd_ROT
0.16024143 0.17575527	0.16024143 0.17575527
0.17575527 0.37881573	0.17575527 0.37881573